REPORT DOCUMENTATION PAGE

AFRL-SR-BL-TR-00-

0633

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
		Final Technical: 7/15/96 - 7/14/99

4. TITLE AND SUBTITLE

Algebraic Geometry and Computational Algebraic Geometry for Image Database Indexing, Image Recognition, And Computer Vision

5. FUNDING NUMBERS

G
F49620-96-1-0363

6. AUTHOR(S)

Dr. Peter Stiller

Texas A&M University
Department of Mathematics
College Station, TX 77843

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Texas A&M Research Foundation
3578 TAMUS
College Station, TX 77843

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFOSR/NM
110 Duncan Ave., Rm-B115
Bolling AFB, DC 20332-8080

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views, opinions and /or findings contained in this report are those of the author(s) and should not be construed as an official Air Force Office of Scientific Research position, policy or decision, unless so designated by other documentation.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This work provides a general framework for the use of geometric invariants in image recognition and computer vision. The theory yields a feature dependent system of equations in variables which represent the 3D invariants of certain features on an object and the 2D invariants of those same features in an image. These equations (called object/image equations) will be satisfied whenever the object produces the image up to suitable transformations of both the object and the image. The significant new contribution of our work is in the use of the theory of correspondences to produce relatively simple "equivariant" polynomial equations to precisely describe these geometric constraints between an object and the images it can produce or an image and the objects that can produce it. A minimal set of generators for the ideal of all such polynomial relations provides an important tool that can be used in recognition algorithms.

20001127 021

14. SUBJECT TERMS	15. NUMBER OF PAGES
object recognition, computer vision, invariants, image understanding, algebraic geometry.	26
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	

DTIC QUALITY INSPECTED 4

N4/Wakman

Final Report
Air Force Office of Scientific Research
Grant No. F49620-96-1-0363
Algebraic Geometry and Computational Algebraic Geometry for Image Database Indexing, Image Recognition, and Computer Vision

Dr. Peter F. Stiller
Department of Mathematics
Texas A&M University
College Station, Texas 77843-3368
stiller@math.tamu.edu

October 29, 1999

A. Summary of Progress - 7/15/96 to 7/14/99

Significant progress was made on all of the theoretical issues. However, as mentioned in our previous progress reports, we encountered some interesting problems related to implementation issues. We devised a number of approaches to solve these problems, and this was the subject of considerable work during the last year of the project. For the most part, the difficulties have been overcome. An additional major theoretical question that we considered centered on methods for database indexing for content based retrieval and geometric hashing. Approaches to this question were addressed in two recent papers, but further improvements in the methods and algorithms are required. Finally, as a result of this research, we are proposing a new paradigm for geometric feature recognition in the full perspective case that shifts the focus to certain toric or generalized toric subvarieties in projective space. The resulting equations should be more global in character and less sensitive to instablity resulting from geometric degeneration and/or noise. With the help of Dr. Greg Arnold at the Air Force Research Laboratory at Wright Patterson Air Force Base, we have done some preliminary work that indicates this approach is feasible and will be considerably more robust than previous methods. We hope to make this the subject of our next proposal to AFOSR.

As previously reported, we have completely determined the object/image equations for configurations of point and line features in the perspective case. The results appear in the papers cited below. We also focused on Grobner basis and sparse resultant techniques (including Dixon and KSY resultants) to provide a symbolic computational approach to generating the object/image equations for various sets of object features and other camera/sensor models. These symbolic computational aspects were discussed in a paper presented at the ISSAC meeting in Hawaii. Grobner bases methods proved intractible - they simply didn't work on computations of the size we were dealing with. The best method proved to be a modified KSY resultant, working modulo large primes, and specializing different subsets of the variables to fixed numerical values. Once the degrees of the variables in the object/image equations were determined, we could interpolate to get the actual equations by taking lots of object/image pairs. So far that hasn't been necessary, as the modified KSY answer has proved to be the right one in characteristic zero. Computational swell is still a problem however. For example, at an intermediate stage in the computation of the object/image equation for the recognition of configurations of 6 line segments, the computation occupied 53MB. However, the final answer collaspes to a relatively small polynomial.

We also made some progress in our effort to explore the feasibility and robustness of several algorithms based on our results (making use of the geometric invariant theory and computational algebraic geometry). These algorithms have been used to index various sorts of databases for content-based retrieval. The computation of the object/image equations is of course a pre-compute, once found for a given feature set they are easily evaluated in real-time to test for object/image matching. The real challenge is to do geometric hashing using the resulting equations when a very large database is involved.

We did develop a multi-dimensional access scheme based on a somewhat complicated hashing technique that works mildly well. For large databases (e.g. 10,000 items) we can do recognition without accessing more than about 15% of the items in worst cases. Moreover the method seems to get more efficient as the size of the database increases. A more advanced method based on a sophisticated polyhedral subdivision scheme is under investigation.

We also have a demo we built in JAVA that will recognize aircraft types in photos. We have a 3D database made using line drawing from Janes. The user selects obvious key points, like the nose, wing tips, etc. in the photo and our algorithms evaluate the object/image equations (using the 3D and 2D invariants) to come up with a good candidate aircraft for the one in the photo. The method is completely view independent. The Sarnoff Corporation in Princeton, New Jersey has an early version of this demo which uses our approach. It has been of interest to the DOD Intellignece community as an aide to photo interpreters. Finally, we have become interested in the potential of our methods for video indexing where dynamic motion is involved.

Most recently we have begun a collaboration with Air Force researchers at Wright Patterson. They are interested in applying our techniques to sequences of images and to other types of imagery, most notably SAR images. Our contacts at Wright Patterson at Drs. Greg Arnold and Vince Velten of AFRL/SNAT (Target Recognition Branch). We hope to submit a proposal based on this collaboration to AFOSR in the near future.

Also, we were put in contact with engineers at Vexcel Corp. in Boulder, CO by Dr. Nachman of AFOSR. We visited Vexcel in summer 1999 to discuss our results.

B. Publications

We have eight papers that have been accepted for publication and others in preparation, inclucing a joint paper with Lewis and Nakos that will likely appear in the Journal of Symbolic Computation. Copies of the two newest published works are attached. Copies of the others were attached to our previous reports.

I. Stiller, P. F., Asmuth, C. A., and Wan, C. S., "Single View Recognition - The Perspective Case," Proceeding SPIE Int'l Conf., Vision Geometry V, Vol. 2826, Denver, CO, 8/96, pp. 226-235 (1996).

2. Stiller, P. F., "General approaches to recognizing geometric configurations from a single view," Proceedings SPIE Int'l Conf, Vision Geometry VI, Vol. 3168, San Diego, CA, 7/97, pp. 262-273 (1997).

3. Stiller, P. F., "Symbolic Computation of Object Image Equations," Proc. of the Int'l Symp. on Symbolic and Algebraic Comp. (ISSAC), Maul, Hawaii, 7/97, pp. 359-364 (1997).

4. Huber, B., Stiller, P. F. and Wan, C. S., "Geometric Content Based Indexing," Proc. of the 26th Workshop on Applied Imagery and Pattern Recognition: Exploiting New Image Sources, Washington, D. C., 6 pages, (1997).

5. Stiller, P. F., "Object Recognition via Configurations of Lines," Proceedings SPIE Int'l Conf, Vision Geometry VII, Vol. 3454, San Diego, CA, 7/98, pp. 76-86 (1998).

6. Huber, B., Stiller, P. F., Wan, C. S., and Shah, T. N., "Invariant Geometric Hashing," Proc. of the 27th Workshop on Applied Imagery and Pattern Recognition, AIPR '98: Advances in Computer Assisted Recognition, Washington, D.C., 10/98, 6 pages (1998).

7. Lewis, R., and Stiller, P. F., "Solving the Recognition Problem for Six Lines Using the Dixon Resultant," Journal of Mathematics and Computers in Simulation, Issue on High Performance Symbolic Computating, Elsevier / North Holland Publishing, Vol. 49, No. 3, pp. 205-219 (1999).

8. Stiller, P. F. and Huber, B., "Geometric Hashing and Object Recognition," Proceedings SPIE Int'l Conf, Vision Geometry VIII, Vol. 3811, Denver, CO, 7/99, pp. 204-211 (1999).

Person completing this report: Peter F. Stiller
Title: Professor of Mathematics and Computer Science
Associate Director Institute for Scientific Computation
Phone: (409) 862-2905
Fax: (409) 845-5827

Date: 29 October 1999

ELSEVIER

Mathematics and Computers in Simulation 49 (1999) 205–219

MATHEMATICS AND COMPUTERS IN SIMULATION

Solving the recognition problem for six lines using the Dixon resultant[1]

Robert H. Lewis[2,a,*], Peter F. Stiller[3,b]

[a]Department of Mathematics, Fordham University, Bronx, NY 10458, USA
[b]Department of Mathematics, Texas A&M University, College Station, TX 77843-3368, USA

Abstract

The "Six-line Problem" arises in computer vision and in the automated analysis of images. Given a three-dimensional (3D) object, one extracts geometric features (for example six lines) and then, via techniques from algebraic geometry and geometric invariant theory, produces a set of 3D invariants that represents that feature set. Suppose that later an object is encountered in an image (for example, a photograph taken by a camera modeled by standard perspective projection, i.e. a "pinhole" camera), and suppose further that six lines are extracted from the object appearing in the image. The problem is to decide if the object in the image is the original 3D object. To answer this question two-dimensional (2D) invariants are computed from the lines in the image. One can show that conditions for geometric consistency between the 3D object features and the 2D image features can be expressed as a set of polynomial equations in the combined set of two- and three-dimensional invariants. The object in the image is geometrically consistent with the original object if the set of equations has a solution. One well known method to attack such sets of equations is with *resultants*. Unfortunately, the size and complexity of this problem made it appear overwhelming until recently. This paper will describe a solution obtained using our own variant of the Cayley–Dixon–Kapur–Saxena–Yang resultant. There is reason to believe that the resultant technique we employ here may solve other complex polynomial systems. © 1999 IMACS/Elsevier Science B.V. All rights reserved.

Keywords: Dixon resultant; Fermat program; Six-Line Problem

1. Introduction

The recognition problem for six lines (Six-Line Problem) arises in computer vision and in the automated recognition of three-dimensional objects. From an object, six lines are extracted, and from those six lines, nine three-dimensional (3D) invariants are computed as a kind of signature. Later, a two-dimensional "snapshot" of some possibly different object is obtained from an arbitrary

* Corresponding author.
[1]Expanded version of talks presented at the Maui IMACS meeting, July 1997, and the Prague IMACS meeting, August 1998.
[2]Partially supported by the Office of Naval Research.
[3]Partially supported by the Air Force Office of Scientific Research.

PII: S0378-4754(99)00050-6

perspective, and from this snapshot six lines are extracted leading to the computation of four two-dimensional (2D) invariants. The question is: Is the snapshot a picture of the original object, i.e. a perspective projection of the original six lines? We desire a method that can rapidly and reliably decide if a given set of 2D data represents the same 3D object, or at least that a given 2D set *cannot* represent that object.

Using algebraic geometry, Stiller [5] showed that there should be a single equation relating the nine 3D invariants to the four 2D invariants. He reduced the problem to a system of four equations in 16 variables involving three additional variables (actually four, but one may be set to 1). The resulting four polynomial equations $d_i = 0, i = 1, \dots, 4$ in the three new variables are quadratic and involve the 9+4=13 invariants as parameters in the coefficients. The image is consistent with the original object if and only if the four equations have a solution in the three variables (subject to a mild nondegeneracy constraint).[4] Note that we do not need to know what the values of the three auxiliary variables actually are, only that a solution exists. Image recognition questions of this general type, but for points, were considered by Quan [4] and Stiller et al. [6].

The solution of systems of polynomial equations is important in many fields of applied mathematics. One of the classic methods of solving such systems is with *resultants*. In general a resultant is a single polynomial derived from a system of polynomial equations that encapsulates the solution (common zeroes) of the system. The *Sylvester Determinant* is the best known method of computing a resultant. However, it is not a realistic tool for solving equations of more than one variable. Other methods exist, which usually compute not the resultant itself but rather a multiple of it, containing *extraneous factors*. The standard Macaulay resultant yields no information for our problem since both the numerator and denominator determinants are identically zero. Another resultant method is that of Dixon (generalizing Cayley), recently extended by Kapur et al. [2]. The authors of that paper show that their method must work if a certain condition holds. The condition is rather strong, and in our case it is not satisfied. Yet we are able to make the method work anyway. This suggests to us that more theoretical work should be done on the Dixon–Kapur–Saxena–Yang approach, and that probably our approach here will succeed for many problems of interest.

2. The basic geometric approach

The moduli space of equivalence classes of (semi-stable) six-tuples of lines in $\mathbb{P}^3$, projective 3-space, under the action of projective transformations (the matrix group PGL_4, 4×4 matrices modulo scalars) is a rational variety of dimension 9. We can thus expect to find 9 functions of the parameters defining the lines which are invariant, in the sense that they provide coordinates on a Zariski open set of the moduli space. We explain briefly how this is done. It is sufficient to work in a Zariski open subset of the set of 6-tuples of lines, so we will not hesitate to impose various general position assumptions that will become apparent below.

Let ℓ_1, ℓ_2, ℓ_3, ℓ_4, ℓ_5, and ℓ_6 be six lines in space. We assume ℓ_1, ℓ_2, and ℓ_3 are mutually skew (our first general position assumption). Without loss of generality, we can complexify and work in complex projective space $\mathbb{P}^3$. Since lines in $\mathbb{P}^3$ are parameterized by the 4-dimensional (complex)

[4] We do not assume homogeneity. Thus, we expect $n+1$ equations in n variables to, in general, not be solvable. The resultant places a constraint on the 13 coefficient parameters that characterizes solvability.

Grassmannian, $G(2,4)$, of two-planes through the origin in (complex) four-space, an ordered six-tuple $(\ell_1,\ldots,\ell_6)$ of lines can be viewed as a point in the 24-dimensional manifold $\hat{X}=G(2,4)\times\cdots\times G(2,4)$. The group PGL_4 of projective linear transformations acts on $\mathbb{P}^3$ sending lines to lines and hence acts on $\hat{X}$ sending a 6-tuple of lines to another 6-tuple. We are interested in the quotient $X=\hat{X}/PGL_4$ of $\hat{X}$ by this action. Since PGL_4 is 15-dimensional, we expect X to have dimension 9. For various technical reasons (in fact to get a good quotient space) we must limit ourselves to an open dense subset, in fact a Zariski open subset, $\hat{U}$ of $\hat{X}$, and construct the quotient $U=\hat{U}/PGL_4$. For example, the requirement that ℓ_1, ℓ_2, and ℓ_3 be mutually skew is one of the conditions defining $\hat{U}$.

Now lines in projective space correspond to planes through the origin in 4-space, and two skew lines correspond to two planes that intersect only in the origin. We can therefore move ℓ_1 to the z, w-plane and ℓ_2 to the x, y-plane by a 4×4 invertible matrix. In this position, ℓ_1 corresponds to the z-axis in space and ℓ_2 corresponds to a line at infinity that meets both the x- and y-axes. Specifically the points $(0:0:1:0)$ and $(0:0:0:1)$ will be on ℓ_1 and likewise, ℓ_2 will contain the points $(1:0:0:0)$ and $(0:1:0:0)$.

Having moved ℓ_1 and ℓ_2 to the above "canonical" positions, the 4×4 invertible matrices that fix these two lines have the form:.

$$M=\begin{pmatrix} a & b & \vdots & 0 & 0\\ c & d & \vdots & 0 & 0\\ \cdots & & & & \\ 0 & 0 & \vdots & e & f\\ 0 & 0 & \vdots & g & h\end{pmatrix} \tag{1}$$

with $ad-bc\neq 0$ and $eh-gh\neq 0$.

Now ℓ_3 is assumed to be skew to both ℓ_1 and ℓ_2. Suppose $(m_1:n_1:r_1:s_1)$ and $(m_2:n_2:r_2:s_2)$ are two distinct points on ℓ_3, which is then the line $\alpha(m_1:n_1:r_1:s_1)+\beta(m_2:n_2:r_2:s_2)=(\alpha m_1+\beta m_2:\alpha n_1+\beta n_2:\alpha r_1+\beta r_2:\alpha s_1+\beta s_2)$ as $(\alpha:\beta)$ runs through all points in $\mathbb{P}^1$. If ℓ_3 were to meet ℓ_1, we would have $\alpha m_1+\beta m_2=0$ and $\alpha n_1+\beta n_2=0$ for some non-trivial (α,β). This can happen if and only if $\det\begin{pmatrix} m_1 & m_2\\ n_1 & n_2\end{pmatrix}=0$. Thus ℓ_3 being skew to ℓ_1 means $\det\begin{pmatrix} m_1 & m_2\\ n_1 & n_2\end{pmatrix}\neq 0$. Likewise ℓ_3 skew to ℓ_2 means $\det\begin{pmatrix} r_1 & r_2\\ s_1 & s_2\end{pmatrix}\neq 0$.

We can choose:

$$\begin{pmatrix} a & b\\ c & d\end{pmatrix}=\begin{pmatrix} m_1 & m_2\\ n_1 & n_2\end{pmatrix}^{-1}$$

and

$$\begin{pmatrix} e & f\\ g & h\end{pmatrix}=\begin{pmatrix} r_1 & r_2\\ s_1 & s_2\end{pmatrix}^{-1}$$

so that the 4×4 matrix (1) above moves ℓ_3 to the line through $(1:0:1:0)$ and $(0:1:0:1)$ without moving ℓ_1 or ℓ_2.

The set of 4×4 matrices fixing ℓ_1, ℓ_2, and ℓ_3 consists of all matrices of the form:

$$\begin{pmatrix} \begin{matrix} a & b \\ c & d \end{matrix} & \vdots & * \\ \cdots & \cdots & \cdots \\ * & \vdots & \begin{matrix} a & b \\ c & d \end{matrix} \end{pmatrix},$$

where $ad-bc\neq0$. In other words, we are reduced to finding invariants for an action of PGL_2 on the remaining three lines.

Assume now that ℓ_4 is skew to ℓ_1 and goes through the points $(\tilde{m}_1 : \tilde{n}_1 : \tilde{r}_1 : \tilde{s}_1)$ and $(\tilde{m}_2 : \tilde{n}_2 : \tilde{r}_2 : \tilde{s}_2)$. Our group, PGL_2, which fixes ℓ_1, ℓ_2, ℓ_3, will act on ℓ_4 as follows:

$$\begin{pmatrix} \begin{matrix} a & b \\ c & d \end{matrix} & \vdots & * \\ \cdots & \cdots & \cdots \\ * & \vdots & \begin{matrix} a & b \\ c & d \end{matrix} \end{pmatrix} \begin{pmatrix} \tilde{m}_1 & \tilde{m}_2 \\ \tilde{n}_1 & \tilde{m}_2 \\ \cdots & \cdots \\ \tilde{r}_1 & \tilde{r}_2 \\ \tilde{s}_1 & \tilde{s}_2 \end{pmatrix}, \quad \det\begin{pmatrix} a & b \\ c & d \end{pmatrix} \neq 0,$$

where we will have $\det\begin{pmatrix} \tilde{m}_1 & \tilde{m}_2 \\ \tilde{n}_1 & \tilde{n}_2 \end{pmatrix} \neq 0$ (because ℓ_4 is skew to ℓ_1). Here the line is represented by a 4×2 matrix whose columns are the homogeneous coordinates of two points on the line. Now without loss of generality, we can assume $\begin{pmatrix} \tilde{m}_1 & \tilde{m}_2 \\ \tilde{n}_1 & \tilde{n}_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

The action yields:

$$\begin{pmatrix} \begin{matrix} a & b \\ c & d \end{matrix} \\ \cdots \\ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \tilde{r}_1 & \tilde{r}_2 \\ \tilde{s}_1 & \tilde{s}_2 \end{pmatrix} \end{pmatrix}$$

which is a new line ℓ going through the two points given by the columns of this 4×2 matrix.

Choosing two different points on ℓ amounts to postmultiplying by an arbitrary invertible 2×2 matrix. We can choose $\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1}$ for this purpose. This means that ℓ can be given by:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ \cdots & \cdots \\ \multicolumn{2}{c}{N} \end{pmatrix},$$

where N is the 2×2 matrix:

$$N=\begin{pmatrix} a & b\\ c & d\end{pmatrix}\begin{pmatrix} \tilde{r}_1 & \tilde{r}_2\\ \tilde{s}_1 & \tilde{s}_2\end{pmatrix}\begin{pmatrix} a & b\\ c & d\end{pmatrix}^{-1}.$$

In other words, the orbit of ℓ is just the orbit of $N=\begin{pmatrix} n_{11} & n_{12}\\ n_{21} & n_{22}\end{pmatrix}$ under conjugation.

The orbits with N a scalar matrix, $N=\begin{pmatrix} \tau & 0\\ 0 & \tau\end{pmatrix}$, are just points, i.e. they are fixed points of the action. The nature of the orbits with N not scalar depends on the Jordan form of N. The possibilities are:

Case 1: $\begin{pmatrix} \lambda & 1\\ 0 & \lambda\end{pmatrix}$. Here the orbit is 2D since the matrices which fix $\begin{pmatrix} \lambda & 1\\ 0 & \lambda\end{pmatrix}$ under conjugation (i.e. commute with $\begin{pmatrix} \lambda & 1\\ 0 & \lambda\end{pmatrix}$ are of the form $\begin{pmatrix} a & b\\ 0 & a\end{pmatrix} a\neq 0$.

Case 2: $\begin{pmatrix} \lambda_1 & 0\\ 0 & \lambda_2\end{pmatrix}$ with $\lambda_1\neq\lambda_2$. Here the orbit is 2D since the matrices which fix $\begin{pmatrix} \lambda_1 & 0\\ 0 & \lambda_2\end{pmatrix}$ under conjugation are of the form $\begin{pmatrix} a & 0\\ 0 & d\end{pmatrix}$.

We will assume that ℓ_4 is in case 2, which is the generic case. In other words, we will assume that $\begin{pmatrix} \tilde{r}_1 & \tilde{r}_2\\ \tilde{s}_1 & \tilde{s}_2\end{pmatrix}$ has distinct (unequal) eigenvalues. Thus we can move ℓ_4 to either the line through $(1:0:\lambda_1:0)$ and $(0:1:0:\lambda_2)$ or the line through $(1:0:\lambda_2:0)$ and $(0:1:0:\lambda_1)$. This ambiguity arises because Jordan form in this case is not unique! It can be either $\begin{pmatrix} \lambda_1 & 0\\ 0 & \lambda_2\end{pmatrix}$ or $\begin{pmatrix} \lambda_2 & 0\\ 0 & \lambda_1\end{pmatrix}$. Now fix ℓ_4 to be one of these two lines. (It does not matter which; moreover we will never in practice need to make a choice between the two.)

The transformations that fix ℓ_1, ℓ_2, ℓ_3 and ℓ_4 take the form:

$$\left(\begin{array}{cc:cc} a & 0 & & \\ & & & *\\ 0 & d & & \\ \hdashline & & a & 0\\ * & & & \\ & & 0 & d\end{array}\right)$$

modulo scalar matrices. Thus we have essentially reduced the group to $\mathbb{C}^*\times\mathbb{C}^*/\mathbb{C}^*\cong\mathbb{C}^*$ where the $\mathbb{C}^*$ in the quotient is embedded diagonally in $\mathbb{C}^*\times\mathbb{C}^*$. We say "essentially", because there is still a $\mathbb{Z}_2$-action lurking that switches $\begin{pmatrix} \lambda_1 & 0\\ 0 & \lambda_2\end{pmatrix}$ and $\begin{pmatrix} \lambda_2 & 0\\ 0 & \lambda_1\end{pmatrix}$. This is accounted for below.

If we assume, in addition to ℓ_4, that ℓ_5 and ℓ_6 are skew to ℓ_1, then we can reinterpret our problem as one of finding invariants for the action of PGL_2 on the three-fold product of 2×2 matrices by conjugation in each factor; specifically:

$$(N_4, N_5, N_6) \rightarrow (AN_4A^{-1}, AN_5A^{-1}, AN_6A^{-1})$$

for A an invertible 2×2 matrix representing an element of PGL_2. Here ℓ_i, $i=4,5,6$ is the line passing through the points in $\mathbb{P}^3$ which are the columns of the 4×2 matrix:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ \multicolumn{2}{c}{\ldots} \\ \multicolumn{2}{c}{N_i} \end{pmatrix}.$$

A known set of invariants are the traces of N_1, N_2, N_3, N_1^2, N_2^2, N_3^2, N_1N_2, N_1N_3, N_2N_3 and $N_1N_2N_3$ which have one relation among them. We take a different approach. Since we have assumed that N_4 has distinct eigenvalues, we can find an A which conjugates N_4 to either $\begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$ or $\begin{pmatrix} \lambda_2 & 0 \\ 0 & \lambda_1 \end{pmatrix}$.

Consider the following subgroup G of PGL_2:

$$G = \left\{ \begin{pmatrix} a & 0 \\ 0 & d \end{pmatrix},\ a \neq 0,\ d \neq 0 \quad \text{or} \quad \begin{pmatrix} 0 & b \\ c & 0 \end{pmatrix},\ b \neq 0,\ c \neq 0 \right\} \text{mod scalars.}$$

Action by G leaves N_4 in diagonal (Jordan) form. Thus we can reduce our action to one of G acting on $(\mathbb{C}\times\mathbb{C}-\Delta)\times N_5 \times N_6$ where Δ is the diagonal in $\mathbb{C}\times\mathbb{C}$ and where we identify (λ_1, λ_2), $\lambda_1 \neq \lambda_2$, in $\mathbb{C}\times\mathbb{C}-\Delta$ with $\begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$.

We now try to move ℓ_5 to a canonical position using just the $\mathbb{C}^*$ action of $\begin{pmatrix} a & 0 \\ 0 & d \end{pmatrix}$ mod scalars. (This does not depend on our choice for the position of ℓ_4.) If we assume ℓ_5 is skew to ℓ_1 so that it can be taken to go through the points $(1:0:n_{11}:n_{21})$ and $(0:1:n_{12}:n_{22})$, then the group acts via:

$$\left(\begin{array}{cc|cc} a & 0 & & \\ & & \vdots & * \\ 0 & d & & \\ \hline & & a & 0 \\ * & \vdots & & \\ & & 0 & d \end{array}\right) \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ n_{11} & n_{12} \\ n_{21} & n_{22} \end{pmatrix} = \begin{pmatrix} a & 0 \\ 0 & d \\ an_{11} & an_{12} \\ dn_{21} & dn_{22} \end{pmatrix}$$

which is the same line as:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ n_{11} & \frac{a}{d}n_{12} \\ \frac{d}{a}n_{21} & n_{22} \end{pmatrix}.$$

We will assume that ℓ_5 is sufficiently generic so that $n_{12} \neq 0$ and $n_{21} \neq 0$. We can then normalize $\begin{pmatrix} n_{11} & n_{12} \\ n_{21} & n_{22} \end{pmatrix}$ so that $n_{12} = n_{21} = g \neq 0$, by choosing $(d/a) = \sqrt{n_{12}/n_{21}}$.

Note that:

$$\begin{pmatrix} 0 & b \\ c & 0 \end{pmatrix} \begin{pmatrix} n_{11} & n_{12} \\ n_{21} & n_{22} \end{pmatrix} \begin{pmatrix} 0 & b \\ c & 0 \end{pmatrix}^{-1} = \begin{pmatrix} n_{22} & \frac{b}{c} n_{21} \\ \frac{c}{b} n_{12} & n_{11} \end{pmatrix}.$$

Thus if we normalize N_5 to $\begin{pmatrix} n_{11} & g \\ g & n_{22} \end{pmatrix}$, $g \neq 0$, then the elements in the subgroup G which preserve our "normal form", namely that N_4 be diagonal and that N_5 have equal off-diagonal elements (non-zero), form a subgroup H:

$$H = \left\{ \begin{pmatrix} a & 0 \\ 0 & a \end{pmatrix}, a \neq 0 \right\} \bigcup \left\{ \begin{pmatrix} a & 0 \\ 0 & -a \end{pmatrix}, a \neq 0 \right\} \bigcup \left\{ \begin{pmatrix} 0 & a \\ a & 0 \end{pmatrix}, a \neq 0 \right\} \bigcup \left\{ \begin{pmatrix} 0 & -a \\ a & 0 \end{pmatrix}, a \neq 0 \right\}$$

mod scalars. Clearly $H < PGL_2$ is a finite group isomorphic to $\mathbb{Z}_2 \times \mathbb{Z}_2$.

We are therefore reduced to the action of this finite group H on $U = (\mathbb{C} \times \mathbb{C} - \Delta) \times (\mathbb{C}^2 \times \mathbb{C}^*) \times \mathbb{C}^4 \subset \mathbb{C}^9$ with coordinates $(\lambda_1, \lambda_2, n_{11}, n_{22}, g, p_{11}, p_{12}, p_{21}, p_{22})$ where ℓ_6 is assumed skew to ℓ_1 so that it can be represented by:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix}.$$

Note that $U \subset \mathbb{C}^9$ is defined by $g \neq 0$ and $\lambda_1 \neq \lambda_2$, i.e. by $g(\lambda_1 - \lambda_2) \neq 0$. Thus U is an affine variety with coordinate ring:

$$R = \mathbb{C}\left[\lambda_1, \lambda_2, \frac{1}{\lambda_1 - \lambda_2}, n_{11}, n_{22}, g, \frac{1}{g}, p_{11}, p_{12}, p_{21}, p_{22}\right]$$

and function field:

$$F = \mathbb{C}(\lambda_1, \lambda_2, n_{11}, n_{22}, g, p_{11}, p_{12}, p_{21}, p_{22}).$$

The desired quotient variety U/H is affine with coordinate ring given by the invariants R^H and function field given by the fixed field F^H. One can show that this variety is rational, i.e. F^H is a field of rational functions in nine algebraically independent quantities – the desired invariants.

To generate the desired equations one works with the nine "invariants" λ_1, λ_2, n_{11}, n_{22}, g, p_{11}, p_{12}, p_{21}, and p_{22} (modulo the action of $H \cong \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$). In the plane one will have four standard invariants q_1, q_2, q_3, q_4 which are rational expressions in the coefficients of the six lines $a_i x + b_i y + c_i = 0$ viewed in $\mathbb{P}^2$ as the points $(a_i : b_i : c_i)$, $i = 1, \ldots, 6$. These are $q_1 = q_{5,0}/q_{5,2}$, $q_2 = q_{5,1}/q_{5,2}$, $q_3 = q_{6,0}/q_{6,2}$, $q_4 = q_{6,1}/q_{6,2}$ in the notation of [6].

Now one can use the above invariants, and the description of the relationship between six lines in 3D and six lines in 2D as a correspondence (in the sense of algebraic geometry), to produce a system of four equations in 17=9+4+4 variables, nine 3D invariants, four 2D invariants, and four variables which represent an invertible 2×2 matrix:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \quad a_{11}a_{22} - a_{21}a_{12} \neq 0$$

acting by conjugation as above on:

$$\left(\begin{pmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{pmatrix}, \begin{pmatrix}n_{11} & g\\ g & n_{22}\end{pmatrix}, \begin{pmatrix}p_{11} & p_{12}\\ p_{21} & p_{22}\end{pmatrix}\right).$$

The key result is that it can be shown that these 3D configurations for fixed $\lambda_1, \lambda_2, \ldots, p_{22}$ and variable a_{ij} sweep out a Zariski open set of the 3D set of all possible 2D equivalence classes obtainable by all possible perspective projections. The resulting four equations appear in Appendix A. Note they are linear in the 2D invariants, quadratic in the 3D invariants and homogeneous quartic in the a_{ij}. By eliminating the a_{ij} one arrives at the desired object/image equation. This is the problem we take up. One complication is that the system always has degenerate solutions a_{ij} where $a_{11}a_{22} - a_{12}a_{21} = 0$. This is what causes the classical Macaulay resultant to fail.

The reader may wonder about the fact that $\lambda_1, \lambda_2, \ldots, p_{22}$ are not quite invariant and that a $\mathbb{Z}_2\times\mathbb{Z}_2$ action still lurks. This causes no serious problem. In fact a test of the final single resultant equation relating $\lambda_1, \ldots, p_{22}, q_1, \ldots, q_4$ shows it to be invariant under this action. For simplicity we stick with these "not quite invariant" invariants.

3. The basic computational approach

We have:

- Nine 3D parameters: $\lambda_1, \lambda_2, n_{11}, n_{22}, g, p_{11}, p_{12}, p_{21}, p_{22}$.
- Four 2D parameters: q_1, q_2, q_3, q_4.
- Three (initially four) conversion variables: $(a_{11} = 1)$, a_{12}, a_{21}, a_{22}.
- Four quartic equations (see Appendix A) in the variables a_{ij} and the 13 parameters.

The four equations have the useful property that q_i appears only in equation i, and only with degree 1. It is therefore quite easy to solve for each q_i in terms of the other variables. While this is an unnatural thing to do from the standpoint of the Six-Line Problem, we will exploit it later to check answers.

The Cayley–Dixon method to eliminate the three variables a_{ij} may be summarized as follows (see [2] for details):

- Adjoin three new auxiliary variables, r, s, t.
- Create the Dixon matrix, DM. Then compute the Dixon polynomial:

$$dm = \frac{\det(DM)}{(r - a_{12})(s - a_{21})(t - a_{22})}.$$

- If desired, we may work with a certain "fixed object," i.e. a set of numerical 3D invariants. Stiller provided an algorithm for creating such test cases of 3D (and corresponding 2D) data sets. The data are integers or rational numbers. We may then substitute into dm some or all of the nine 3D numerical values. This reduces the size and complexity of dm.
- Create the second Dixon matrix by extracting coefficients from dm in a certain way. These coefficients are polynomials in the four 2D parameters q_i, $i = 1, \ldots, 4$ and those 3D parameters that remain from the previous step. It is a 105×105 matrix.

- The determinant of this second matrix is the classical Dixon Resultant. If there is a common solution of the original system of four equations, then this determinant must be 0. Ideally that provides an equation that must be satisfied by the parameters. However, in our case (and in many others) it is identically 0.

But that is not the end of the story. The Kapur–Saxena–Yang (KSY) method continues:

- Extract the non-zero rows and columns from the second matrix. This leaves a 51×56 matrix. Call this the *third matrix*.
- If a certain condition holds on the third matrix, compute the determinant of any maximal rank submatrix. These polynomials must vanish if the original system has a solution.

In other words, these necessarily nonzero polynomials, any of which we will call *ksy*, play the role of the classical Dixon Resultant. We will have more to say about the "certain condition" in Section 5.

4. Phase One of the computation

Unless some of the numerical 3D parameters from a "real" object are substituted into *dm* before the creation of the second Dixon matrix, the polynomials *ksy* will be hopelessly large for any existing computer system.[5] In the first phase of the project, we substituted rational values for all nine 3D parameters, thus reducing the goal to computing a resultant for that one object – a polynomial in the four 2D invariants q_1, q_2, q_3, q_4.

- Input the numerical (rational or integral) data for the nine 3D parameters. For example λ_1, λ_2, ..., p_{22}=3, 4, 2, 3, 2, 3,1, 2, 1/2.
- Compute *ksy*, a polynomial in the four parameters q_1, q_2, q_3, q_4.
- To determine if a set of 2D data q_1, q_2, q_3, q_4 "matches" the 3D object, substitute the four numerical values into *ksy* and see if the result is 0. If it is not, the 2D set cannot be a perspective projection of the 3D object.

An important simplification results by reconsidering what is meant by "the result is 0". Recall that the coefficients of the polynomial *ksy* are rational numbers. Since we seek solutions of $ksy=0$, we can clear out denominators and assume that all coefficients are integers. Rather than work over the ring of integers, we can save enormously in both time and space if we choose a moderately large (20,000–50,000) prime number p at random and reduce all the equations modulo this prime. We are then working over the field $\mathbb{Z}_p$, and it is sufficient to test a candidate set of 2D parameters s_1, s_2, s_3, s_4 by reducing them modulo p and checking $ksy(s_1, s_2, s_3, s_4) = 0$ in $\mathbb{Z}_p$. The resulting modular algorithm is probabilistic, with a very high probability of success. An incorrect set of parameters will not pass the modular test unless $ksy(s_1, s_2, s_3, s_4)$ is a multiple of p. The probability of that is no more than $1/p$. A correct set of parameters will pass it unless one of the parameters is a fraction with denominator a multiple of p. Since p will be around 20,000, this seems extremely unlikely, and in any event would be detected when the rational values s_1, s_2, s_3, s_4 are converted to values in $\mathbb{Z}_p$. The probability of a mistaken judgement can be further reduced by simply doing the algorithm twice with two different primes.

Lewis wrote programs to create the third matrix and compute *ksy* in his computer algebra system *Fermat* [3]. One method is to compute the product of the pivot elements that come up as one normalizes

[5]Conservatively, such a *ksy* would have at least 10^9 terms, probably more like 10^{12}.

(say, into the Hermite form) the third matrix. One can learn the rank of this matrix very easily by plugging in integers at random for the four q_i parameters and computing a matrix normal form. The matrix has rank 26. Therefore, *ksy* is the product of 26 terms that will appear on the main diagonal as the matrix is normalized. Depending on the algorithm, these terms may not be all polynomials. Nevertheless, the product of all 26 will be a polynomial.

The row and column reductions went well, up to the 17th row/column. Beyond that the complexity of the computation becomes overwhelming. However, it is not necessary to continue the normalization algorithm. Recall that we have reason to think that any maximal rank submatrix will do. By substituting random integers for three of the parameters q_i it is easy to discover a 26×26 maximal rank submatrix. *ksy* is just the determinant of this fourth matrix. Since all its entries are (4 variable) polynomials, the determinant algorithm in *Fermat* (there are several) which works by recursive Lagrange interpolation is suitable. It completed in 3 h and produced a *ksy* with around 500,000 terms. (All times in this paper are for a 233 mHz Macintosh with 604e chip.) It had degree 26 or 25 in each of the four q_i. As an ASCII file, this *ksy* occupied a file of 3.5 Mb.

To evaluate *ksy* at four numerical values took about 2 s, so this is feasible in real time. Extensive testing with 2D data sets, valid and invalid, verified the correctness of *ksy*. This was all done using the prime 44,449. Using 41,999 produced essentially the same results.

Wishing to look more closely at *ksy*, we returned to the idea of computing it by row reductions on the third matrix, over the field $\mathbb{Q}$, rather than $\mathbb{Z}_p$. The first nine diagonal pivot elements were enlightening:

$$q_4-q_2,\ q_4-q_2,\ q_4-q_2,\ q_2(q_4+q_2),\ q_2(q_4+q_2),\ (q_3+q_1)(q_4-q_2),$$
$$(q_3-q_1)(q_4-q_2),\ (q_3-2q_1)(q_4-q_2),\ (q_3-1/2q_1+1/6)(q_4-q_2).$$

This suggests, but does not prove, that *ksy* has many simple factors. After much testing Lewis verified that:

$$q_4(q_4-q_2)^4(q_3-q_1)^4q_2^2(q_3-1/2q_1+1/6) \tag{2}$$

is a factor. One of the *Fermat* determinant algorithms can take advantage of a known factor. It then computed the rest of *ksy* (the other factor) in only 25 min, down from the original 3 h. This "reduced *ksy*" has 100,000 terms and occupies only 670 K of disk space. Numerical tests show that the actual resultant is indeed a factor of the reduced *ksy*.

Even more extraneous factors can be removed from the reduced *ksy*. First, since the resultant must be irreducible, we may divide out all the contents of *ksy*. Secondly, with different maximal rank submatrices, simple variations of (1) divide their determinants (and this remains true for different choices of 3D invariants, not just the values used here, 3, 4, 2, 3, 2, 3, 1, 2, 1/2). Thus, it is not hard to compute another reduced determinant *ksy′*, and the true resultant should be a factor of *GCD(ksy, ksy′)*. We have therefore the following algorithm:

```
res:=ksy;
REPEAT
   Compute new reduced ksy using a new maximal rank submatrix;
   ksy:=ksy/all contents(ksy);
   res:=GCD(res, ksy)
UNTIL DONE
```

After five repetitions of this loop, the polynomial *res* contained only 300 terms! It was small enough to be factored with standard algorithms. The factor that vanishes on a known 2D data set is:

$$\begin{aligned} sixline = {} & q_1^2q_4^2 - 2q_1q_4^2 + 8q_4^2 + 6q_2q_3q_4 + 12q_1q_3q_4 - 60q_3q_4 - 2q_1q_2q_4 + 2q_1^2q_4 + 28q_1q_4 \\ & - q_2^2q_3^2 + 8q_3^2 - 2q_2^2q_3 - 14q_1q_2q_3 + 60q_2q_3 - 16q_1q_3 - 8q_2^2 - 28q_1q_2 + 8q_1^2. \end{aligned}$$

This was all done over a finite field, $\mathbb{Z}_p$. But the coefficients above are suggestively small integers. Indeed, *this is the actual resultant over* $\mathbb{Q}$, *not just over* $\mathbb{Z}_p$. That is easy to prove: recall that each q_i may be solved for in $d_i = 0$, then just substitute into *sixline* each q_i with its formula in terms of the other variables. The expression evaluates to 0. It is as if we had set out to use the Chinese Remainder Theorem to find the resultant over $\mathbb{Q}$, and discovered that one prime was enough.

In summary, the polynomial *sixline* provides the solution to the problem for the given particular 3D data set. If any set of 2D invariants be presented in the future, plug them into *sixline*. If the result is not 0, then they do not represent a perspective projection of the original object.

Now, our entire method, which we know has worked because *sixline* is verifiably correct, is based on the Kapur–Saxena–Yang idea of computing the determinant of a maximal rank submatrix. In [2] they show that the resultant must be a factor of any such determinant, provided that a certain condition holds. This (sufficient) condition is that some column in the 105×105 second matrix be linearly independent of all the others. However, in our case the condition fails! Yet the method works anyway.

It may be asked why it was necessary to produce the polynomial *sixline* at all. Instead, one could simply take a candidate set of 2D invariants and plug them into the third matrix, whose rank is known to be 26. If the rank drops, which is surely a simple thing to check, then the determinant of every maximal rank submatrix must vanish on that 2D set.

To answer, there are several reasons why the derivation of the polynomial *sixline* is very desirable:

- It is not clear that the g.c.d. of all the maximal rank submatrices is exactly the resultant. If it is not, there may be spurious zeros.
- The 2D invariants $\{q_1, q_2, q_3, q_4\}$ will probably be obtained by extracting and measuring lines on photographs. It is necessary to match the six 2D lines with the six lines on the original 3D object. This will probably require testing all 6!=720 possible permutations. The time saved in plugging the $\{q_i\}$ into *sixline* versus finding the rank of the third matrix may not be significant, but it will be multiplied by 720.
- We have been assuming that the 2D invariants are known exactly, but if they come from measurements, there may be errors. Error analysis is much easier if it is based on the polynomial *sixline*.
- In the following sections we generalize our method to produce a completely symbolic version of *sixline*; i.e., we forego plugging in numerical values for the nine 3D parameters, and all 13 variables appear in the resultant.
- It is possible to consider recognition of n lines by similar methods. However six is the minimum for the problem to be meaningful; sets of five or fewer lines cannot be distinguished in this manner.

5. Phase Two

In Phase One we substituted numerical values for all parameters except q_1, q_2, q_3, q_4. Lewis then redid the computations keeping various other subsets of the parameters, such as the four p_{ij}, the set

$\{\lambda_1, \lambda_2, g, n_{11}, n_{22}\}$, and various combinations of the preceding with some of the q_i. In this way we learned the degree of the resultant in all of its parameters. Each degree is either 1 or 2. We learned also that if we order the parameters so that the four q_i have highest precedence, the leading term is $f(\lambda_1, \lambda_2, p_{11}, p_{22}, p_{12}, p_{21})q_1^2q_4^2$, for some polynomial f in the indicated parameters only.

6. Phase Three

The work done in Phase One constitutes a viable solution to the Six-Line Problem, given the 3D data of an object. But we want to compute the complete resultant for all objects, in all 13 parameters.

Grosshans et al. [1] were the first to compute this polynomial *res*, using invariant theory and experimenting with lots of numerical cases, observing various dependencies among the variables and exploiting various symmetries in the equation. They found a *res* with 239 terms. The final answer is quartic in the 3D invariants and quartic in the 2D invariants, yielding a total degree 8. An alternative approach by Stiller and Ma used interpolation by generating a large number of "matching" object image pairs and exploiting the degree bounds predicted by Lewis. How do we know this polynomial is correct? Recall that each q_i occurs only in equation $d_i = 0$ and can be solved for, yielding a rational expression, for example:

$$
\begin{aligned}
q_1 = (&g\lambda_2a_{22}^3 - g\lambda_1a_{12}a_{21}a_{22}^2 - n_{22}\lambda_2a_{21}a_{22}^2 + n_{11}\lambda_2a_{21}a_{22}^2 + 2g\lambda_2a_{12}a_{22}^2 \\
&-g\lambda_1a_{12}a_{22}^2 - n_{11}n_{22}\lambda_2a_{22}^2 + n_{11}\lambda_2a_{22}^2 + g^2\lambda_2a_{22}^2 + n_{22}\lambda_1a_{12}a_{21}^2a_{22} - n_{11}\lambda_1a_{12}a_{21}^2a_{22} \\
&-g\lambda_2a_{21}^2a_{22} - g\lambda_1a_{12}^2a_{21}a_{22} + n_{11}n_{22}\lambda_2a_{12}a_{21}a_{22} \\
&-2n_{22}\lambda_2a_{12}a_{21}a_{22} + n_{11}\lambda_2a_{12}a_{21}a_{22} - g^2\lambda_2a_{12}a_{21}a_{22} + n_{11}n_{22}\lambda_1a_{12}a_{21}a_{22} \\
&+n_{22}\lambda_1a_{12}a_{21}a_{22} - 2n_{11}\lambda_1a_{12}a_{21}a_{22} - g^2\lambda_1a_{12}a_{21}a_{22} - g\lambda_2a_{21}a_{22} \\
&+g\lambda_2a_{12}^2a_{22} - g\lambda_1a_{12}^2a_{22} - n_{11}n_{22}\lambda_2a_{12}a_{22} + n_{11}\lambda_2a_{12}a_{22} + g^2\lambda_2a_{12}a_{22} \\
&+n_{11}n_{22}\lambda_1a_{12}a_{22} - n_{11}\lambda_1a_{12}a_{22} - g^2\lambda_1a_{12}a_{22} + g\lambda_1a_{12}a_{21}^3 - n_{11}n_{22}\lambda_1a_{12}^2a_{21}^2 \\
&+n_{22}\lambda_1a_{12}^2a_{21}^2 + g^2\lambda_1a_{12}^2a_{21}^2 - g\lambda_2a_{12}a_{21}^2 + 2g\lambda_1a_{12}a_{21}^2 + n_{11}n_{22}\lambda_2a_{12}^2a_{21} \\
&-n_{22}\lambda_2a_{12}^2a_{21} - g^2\lambda_2a_{12}^2a_{21} - n_{11}n_{22}\lambda_1a_{12}^2a_{21} + n_{22}\lambda_1a_{12}^2a_{21} + g^2\lambda_1a_{12}^2a_{21} \\
&- g\lambda_2a_{12}a_{21} + g\lambda_1a_{12}a_{21})/(g\lambda_2a_{12}a_{21}a_{22}^2 - g\lambda_1a_{12}a_{21}a_{22}^2 - n_{22}\lambda_2a_{21}a_{22}^2 + n_{22}\lambda_1a_{21}a_{22}^2 \\
&+ g\lambda_1\lambda_2a_{12}a_{22}^2 - g\lambda_1a_{12}a_{22}^2 - n_{22}\lambda_1\lambda_2a_{22}^2 + n_{22}\lambda_1a_{22}^2 \\
&+n_{11}\lambda_2a_{12}a_{21}^2a_{22} - n_{11}\lambda_1a_{12}a_{21}^2a_{22} - g\lambda_2a_{21}^2a_{22} + g\lambda_1a_{21}^2a_{22} \\
&-g\lambda_1\lambda_2a_{12}^2a_{21}a_{22} + 2g\lambda_2a_{12}^2a_{21}a_{22} - g\lambda_1a_{12}^2a_{21}a_{22} + n_{22}\lambda_1\lambda_2a_{12}a_{21}a_{22} \\
&+n_{11}\lambda_1\lambda_2a_{12}a_{21}a_{22} - 2n_{22}\lambda_2a_{12}a_{21}a_{22} + n_{11}\lambda_2a_{12}a_{21}a_{22} + n_{22}\lambda_1a_{12}a_{21}a_{22} \\
&-2n_{11}\lambda_1a_{12}a_{21}a_{22} - g\lambda_1\lambda_2a_{21}a_{22} - g\lambda_2a_{21}a_{22} + 2g\lambda_1a_{21}a_{22} \\
&+g\lambda_1\lambda_2a_{12}^2a_{22} - g\lambda_1a_{12}^2a_{22} - n_{22}\lambda_1\lambda_2a_{12}a_{22} + n_{11}\lambda_1\lambda_2a_{12}a_{22}
\end{aligned}
$$

$$+n_{22}\lambda_1 a_{12}a_{22} - n_{11}\lambda_1 a_{12}a_{22} - g\lambda_1\lambda_2 a_{22} + g\lambda_1 a_{22} - n_{11}\lambda_1\lambda_2 a_{12}^2 a_{21}^2$$
$$+n_{11}\lambda_2 a_{12}^2 a_{21}^2 + g\lambda_1\lambda_2 a_{12}a_{21}^2 - g\lambda_2 a_{12}a_{21}^2 - g\lambda_1\lambda_2 a_{12}^3 a_{21} + g\lambda_2 a_{12}^3 a_{21}$$
$$+n_{22}\lambda_1\lambda_2 a_{12}^2 a_{21} - n_{11}\lambda_1\lambda_2 a_{12}^2 a_{21} - n_{22}\lambda_2 a_{12}^2 a_{21} + n_{11}\lambda_2 a_{12}^2 a_{21}$$
$$+g\lambda_1\lambda_2 a_{12}a_{21} - g\lambda_2 a_{12}a_{21})$$

Lewis simply substituted for each q_i its expression as above into *res*·and checked that the result is identically (symbolically) 0. (200 meg of RAM, 11 min, using *Fermat*. No other computer algebra system that we are aware of could do this computation.)

We felt strongly that the Dixon–KSY method ought to work as well to compute *res*. But recall that even after plugging in integers for 9 of the 13 parameters, the KSY method produced a 500,000 term answer, almost all of which were spurious factors. Brute force is therefore rejected. Several ideas led eventually to the solution.

The first idea, due to George Nakos, is as follows. Instead of applying the KSY method to four equations $\{d_i = 0\}$ to eliminate the three variables $\{a_{12}, a_{21}, a_{22}\}$, do it in stages:

- Apply KSY to $\{d_1, d_2, d_3\}$ eliminating two variables, obtaining a polynomial y_1 that still has a_{12}.
- Apply KSY to $\{d_2, d_3, d_4\}$ eliminating two variables, obtaining a polynomial y_2 that still has a_{12}.
- Apply KSY to $\{y_1, y_2\}$ to get the final *res*.

However, it is not that easy. Each y_i would have had many millions of terms, making the third step hopeless. Lewis applied two fairly standard ideas to reduce the size of each y_i.

- *Interpolation*: Plug in authentic 3D values for some of the parameters. Run the above three steps with enough such sets of values, then construct *res* with standard interpolation techniques.
- *Quotient ring*: We know that the final answer *res* is of low degree in each parameter; for example, it is degree 2 in g and degree 1 in n_{11}. Analogously to working over $\mathbb{Z}_p$ instead of $\mathbb{Z}$, we could work modulo a cubic polynomial in g and a quadratic polynomial in n_{11}. This eliminates high degree (in g and n_{11}) intermediate results while, ideally, not changing the final answer. *Fermat* allows one to work easily and efficiently over such fields.

While either technique alone might have sufficed, we decided to use both. There is a problem, however, with the second technique, the well known *leading coefficient problem*. Suppose R is a polynomial ring, say $R = F[a, b, c, \ldots]$. Let $I \subset R$ be an ideal such that R/I is a field. We wish to compute in $(R/I)[x, y, z, \ldots]$ instead of $R[x, y, z, \ldots]$. When working over such a quotient field, algorithms such as polynomial g.c.d. dispense with leading coefficients involving the field variables a, b, c, …. The leading coefficients are divided through to produce "pseudo-monic" polynomials. This makes reconstruction of the actual answer in $R[x, y, z, \ldots]$ problematic. But due to the work accomplished in Phase Two, we know that the leading term relative to the q_i is $f(\lambda_1, \lambda_2, p_{11}, p_{22}, p_{12}, p_{21})q_1^2 q_4^2$, for some polynomial f in the indicated parameters. Therefore, by choosing to mod out by g and n_{11} we avoid this problem. (We could mod out by n_{22} in addition, but that greatly slows down the computations in *Fermat*.)

In summary, we chose to work modulo g^3-3 and $n_{11}^2 - 7$, and over the prime p=17041. $\mathbb{Z}_p[g, n_{11}]/\langle g^3 - 3, n_{11}^2 - 7\rangle$ is a field. We interpolated for λ_1, λ_2 and n_{22}. We know from Phase Two that the answer is of degree 1 in each of the latter parameters, so we need to run the three steps eight times.

However, it still does not work. y_1 and y_2 each have about 300,000 terms and, worse yet, are of high degree ($\geqslant 30$) in a_{12}. That makes the third step unworkable. The problem is solved by recalling from Phase One the idea of dividing out by the contents. Compute y_1 (12 min). Then compute all its contents and divide out by them (123 min; 132 meg RAM). The result has only 90 terms! Repeat for y_2. Then do the third step (about 1 min). This produces a preliminary answer with a set of values plugged in for λ_1, λ_2 and n_{22}. We then repeat seven times and interpolate for the final answer. In doing so, one final problem arises. Because the contents were divided out often, the eight preliminary answers may be missing leading numerical coefficients – another incarnation of the leading coefficient problem. Especially likely is that one or more needs to be multiplied by -1. Since we know that the final answer is a polynomial with integer coefficients, it is easy to experiment and compute the right answer.

7. Conclusion

Elimination in stages using the Cayley–Dixon–Kapur–Saxena–Yang method succeeded for two reasons:

1. The final answer is of low degree in most of its variables (in fact, all of them).
2. At each stage, polynomials are produced that are multiples of the resultant, with huge spurious factors. *But the resultant is the only factor involving all the variables.* It can therefore be found by dividing out all the contents.

Unless there is something very special about the equations that came up in this problem, it is reasonable to conjecture that our successive elimination method with KSY may be applicable to other large polynomial systems.

Acknowledgements

We wish to thank Robert M. Williams (NAWCAD, Patuxent River, Maryland) for championing this problem and being a steady source of support, George Nakos (US Naval Academy) for teaching us the Dixon Resultant, and Michael Hirsch (University of Delaware) for helping us in the early stages of the project. We also wish to acknowledge Williams, Ron Gleeson (College of New Jersey), and Frank Grosshans (Westchester of Pennsylvania) for their many contributions to this problem and their work in first solving the system in Appendix A.

Appendix A

A. The four equations

- Nine 3D parameters: λ_1, λ_2, n_{11}, n_{22}, g, p_{11}, p_{12}, p_{21}, p_{22}.
- Four 2D parameters: q_1, q_2, q_3, q_4.
- Four conversion variables (later we set $a_{11}=1$): a_{11}, a_{12}, a_{21}, a_{22}.

- Four equations d_1, d_2, d_3, d_4 in the three variables a_{ij} and the 13 parameters. Note that q_i appears only in d_i and only with exponent 1.

$$\begin{aligned}
&d_1 = (ga_{11}^2 + ga_{11}a_{21} + n_{22}a_{11}a_{12} + a_{11}a_{22}n_{22} - n_{11}a_{11}a_{12} - a_{12}a_{21}n_{11}\\
&-ga_{12}^2 - ga_{12}a_{22})(-\lambda_2 a_{12}a_{21} - \lambda_2 a_{21}a_{22} + \lambda_1 a_{11}a_{22} + \lambda_1 a_{21}a_{22} - \lambda_1\lambda_2 a_{11}a_{22}\\
&+\lambda_1\lambda_2 a_{12}a_{21})q_1 - (-ga_{11}a_{21} - ga_{21}^2 - n_{22}a_{12}a_{21} - n_{22}a_{21}a_{22} + n_{11}a_{11}a_{22}\\
&+n_{11}a_{21}a_{22} + ga_{12}a_{22} + ga_{22}^2 - a_{11}a_{22}n_{11}n_{22} + a_{11}a_{22}g^2 + a_{12}a_{21}n_{11}n_{22}\\
&-a_{12}a_{21}g^2)(\lambda_2 a_{11}a_{12} + a_{11}a_{22}\lambda_2 - \lambda_1 a_{11}a_{12} - a_{12}a_{21}\lambda_1),\\
&d_2 = (ga_{11}^2 + ga_{11}a_{21} + n_{22}a_{11}a_{12} + a_{11}a_{22}n_{22} - n_{11}a_{11}a_{12} - a_{12}a_{21}n_{11}\\
&-ga_{12}^2 - ga_{12}a_{22})(a_{11}a_{22} - a_{12}a_{21} - a_{11}a_{22}\lambda_2 + a_{12}a_{21}\lambda_1)q_2 - (a_{11}a_{22} - a_{12}a_{21} - ga_{11}a_{21}\\
&-a_{11}a_{22}n_{22} + a_{12}a_{21}n_{11} + ga_{12}a_{22})(\lambda_2 a_{11}a_{12} + a_{11}a_{22}\lambda_2 - \lambda_1 a_{11}a_{12} - a_{12}a_{21}\lambda_1),\\
&d_3 = (p_{12}a_{11}^2 + p_{12}a_{11}a_{21} + p_{22}a_{11}a_{12} + a_{11}a_{22}p_{22} - p_{11}a_{11}a_{12} - a_{12}a_{21}p_{11}\\
&-p_{21}a_{12}^2 - p_{21}a_{12}a_{22})(-\lambda_2 a_{12}a_{21} - \lambda_2 a_{21}a_{22} + \lambda_1 a_{11}a_{22} + \lambda_1 a_{21}a_{22} - \lambda_1\lambda_2 a_{11}a_{22}\\
&+\lambda_1\lambda_2 a_{12}a_{21})q_3 - (-p_{12}a_{11}a_{21} - p_{12}a_{21}^2 - p_{22}a_{12}a_{21} - p_{22}a_{21}a_{22} + p_{11}a_{11}a_{22}\\
&+p_{11}a_{21}a_{22} + p_{21}a_{12}a_{22} + p_{21}a_{22}^2 - a_{11}a_{22}p_{11}p_{22} + a_{11}a_{22}p_{12}p_{21} + a_{12}a_{21}p_{11}p_{22}\\
&-a_{12}a_{21}p_{12}p_{21})(\lambda_2 a_{11}a_{12} + a_{11}a_{22}\lambda_2 - \lambda_1 a_{11}a_{12} - a_{12}a_{21}\lambda_1),\\
&d_4 = (p_{12}a_{11}^2 + p_{12}a_{11}a_{21} + p_{22}a_{11}a_{12} + a_{11}a_{22}p_{22} - p_{11}a_{11}a_{12} - a_{12}a_{21}p_{11}\\
&-p_{21}a_{12}^2 - p_{21}a_{12}a_{22})(a_{11}a_{22} - a_{12}a_{21} - a_{11}a_{22}\lambda_2 + a_{12}a_{21}\lambda_1)q_4 - (a_{11}a_{22}\\
&-a_{12}a_{21} - p_{12}a_{11}a_{21} - a_{11}a_{22}p_{22} + a_{12}a_{21}p_{11} + p_{21}a_{12}a_{22})(\lambda_2 a_{11}a_{12} + a_{11}a_{22}\lambda_2\\
&-\lambda_1 a_{11}a_{12} - a_{12}a_{21}\lambda_1).
\end{aligned}$$

References

[1] F. Grosshans, R. Gleason, R. Williams, M. Hirsch, August 1997, personal communication.

[2] D. Kapur, T. Saxena, L. Yang, Algebraic and geometric reasoning using Dixon resultants, Proceedings of the International Symposium on Symbolic and Algebraic Computation, 1994, ACM Press, New York.

[3] R.H. Lewis, Computer algebra system *Fermat*. http: //www.bway.net/~lewis/, http://www.fordham.edu/lewis/, 1998.

[4] L. Quan, Computation of the invariants of a point set in P^3 from its projections in P^2, in: N.L. White (Ed.), Invariant Methods in Discrete and Computational Geometry, 1994, pp. 223–244.

[5] P. Stiller, Symbolic computation of object/image equations, Proceedings of the International Symposium on Symbolic and Algebraic Computation, 1997, ACM Press, New York, pp. 359–364.

[6] P. Stiller, C. Asmuth, C. Wan, Single-view recognition – the perspective case, Proceedings SPIE International Conference, Vision Geometry V 2826, vol. 8, 1996, pp. 226–235.

Geometric Hashing and Object Recognition

Peter Stiller and Birkett Huber

Institute for Scientific Computation
Texas A&M University
College Station, TX 77843–3404 USA

ABSTRACT

We discuss a new geometric hashing method for searching large databases of 2D images (or 3D objects) to match a query built from geometric information presented by a single 3D object (or single 2D image). The goal is to rapidly determine a small subset of the images that potentially contain a view of the given object (or a small set of objects that potentially match the item in the image). Since this must be accomplished independent of the pose of the object, the objects and images, which are characterized by configurations of geometric features such as points, lines and/or conics, must be treated using a viewpoint invariant formulation. We are therefore forced to characterize these configurations in terms of their 3D and 2D geometric invariants. The crucial relationship between the 3D geometry and its "residual" in 2D is expressible as a correspondence (in the sense of algebraic geometry). Computing a set of generating equations for the ideal of this correspondence gives a complete characterization of the view independent relationships between an object and all of its possible images. Once a set of generators is in hand, it can be used to devise efficient recognition algorithms and to give an efficient geometric hashing scheme. This requires exploiting the form and symmetry of the equations. The result is a multidimensional access scheme whose efficiency we examine. Several potential directions for improving this scheme are also discussed. Finally, in a brief appendix, we discuss an alternative approach to invariants for generalized perspective that replaces the standard invariants by a subvariety of a Grassmannian. The advantage of this is that one can circumvent many annoying general position assumptions and arrive at invariant equations (in the Plücker coordinates) that are more numerically robust in applications.

1. INTRODUCTION

Content-based retrieval of information from large databases that keys on visual/geometric information contained in images, schematics, design drawings, and geometric models of environments, mechanical parts or molecules, etc., will play an increasingly important role in future distributed information and knowledge systems. This paper focuses on two aspects of geometric content-based retrieval for knowledge acquisition. The first concerns geometric hashing techniques for matching geometric configurations of features in a database of 3D objects to a geometric configuration of features in a single 2D image or, vice versa, matching geometric configurations of features in a database of 2D images to a geometric configuration of features on a single 3D object. The second, dealt with in an appendix, describes an alternative to classical invariants that associates to a particular geometric configuration an invariant subvariety of a Grassmannian. Equations for this subvariety in the Plücker coordinates of the Grassmannian serve as "invariants" of the configuration. The advantage of these "invariants" is that one can circumvent many annoying general position assumptions; resulting in more numerically robust versions of the object/image equations used to match 3D and 2D configurations of features.

2. THE SET-UP

In[1,2] we describe a technique for using geometric information contained in 2-D images to search large databases of 3-D models in the special case where the geometric information consists of finite point configurations. This technique exploits certain polynomial relations known as object/image equations between invariants assigned to the 2-D and 3-D feature sets. The resulting scheme is independent of changes in scale and perspective. Here, we describe a technique for constructing a hashing scheme based on this technology.

Recall that the object/image equations are polynomial relations in the combined set of geometric invariants associated to a 3D configuration and those associated to a configuration in a 2D image. They completely describe

Other author information: (Send correspondence to P.F.S.)
P.F.S.: Email: stiller@isc.tamu.edu; Telephone: 409-862-2905; Fax: 409-845-5827

the mutual 3D/2D constraints[2,3,4], and can be used in a number of ways. For example, from a given 2D configuration one can determine a set of non-linear polynomial constraints on the geometric invariants of those 3D configurations capable of producing the given 2D configuration in an image. This results in a test for limiting the possibilities for the object is being viewed. When applied to searching a database of objects to find likely matches to a given image, this test can then be used as a filter to remove objects from further consideration, leading to at a greatly reduced set of objects which must be more carefully considered. While the object/image formalism applies to much more general feature sets,[5] the case of point configurations is especially simple. Its performance for single view recognition of point configurations was investigated in.[1] In this paper we describe the use of these equations for setting up an index (hashing scheme)into a large data base of 3D point configurations for rapid query by $2D$ images. In order to rule out groups of objects the index uses the object/image equations to determine if *any* object in a region of the "invariants space" for objects matches a given image.

We will work in the general perspective case. To facilitate this, ordinary points $(x, y, z) \in \mathbb{R}^3$ or $(u, v) \in \mathbb{R}^2$ are represented in homogeneous coordinates in the respective projective space $\mathbb{P}^3$ or projective plane $\mathbb{P}^2$:

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \in \mathbb{P}^3 \quad \text{or} \quad \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \in \mathbb{P}^2$$

Image formation is accomplished by a general perspective projection M, which is a 3×4 matrix of rank 3. Viewed as a rational map from $\mathbb{P}^3$ to $\mathbb{P}^2$, M takes (x, y, z) to (u, v) where $u = \frac{\bar{u}}{\bar{w}}$ and $v = \frac{\bar{v}}{\bar{w}}$ and

$$\begin{pmatrix} \bar{u} \\ \bar{v} \\ \bar{w} \end{pmatrix} = M \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}.$$

This transformation is undefined at one point $(\alpha : \beta : \gamma : \delta) \in \mathbb{P}^3$, which is the focal point of the projection (the null space of M). A typical example is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

which takes (x, y, z) to (u, v) where

$$u = \frac{x}{z+1} \quad v = \frac{y}{z+1}$$

and which is undefined at $(0 : 0 : -1 : 1)$. This is perspective projection onto the xy-plane through the point $(0, 0, -1)$. Such projections are examples of the standard "pinhole camera" model of image formation.[6]

In general, we will have a parameter space of point configurations in 3D or 2D. These features will be subject to general projective transformations (the group PGL_4 in space or PGL_3 in the plane) resulting in a notion of equivalent configurations. This physically reflects a "change of point of view" in $\mathbb{P}^3$ or in $\mathbb{P}^2$. The basic invariants we construct are functions of the configurations that remain unchanged under a projective transformation. They provide coordinates on a portion of the space of equivalence classes of configurations. In practice, one must deal with several of these "coordinate patches". (To avoid "coordinate patches" by using more "global" invariants, see the appendix.)

3. THE OBJECT-IMAGE EQUATIONS

With the 3D and 2D invariants in hand, we can construct the basic equations between them that reflect the complete set of view independent constraints that must hold between 3D configurations and their images in 2D under perspective projection. It will be convenient to represent configurations of m points in $\mathbb{P}^d$ by $(d+1) \times m$ matrices $A = [P_1| \ldots |P_m]$, whose columns are homogeneous coordinates for the points P_i. As homogeneous coordinates are defined only up to scalar multiples, we can multiply A on the right by an arbitrary invertible $m \times m$ diagonal matrix without changing the point configuration it represents. We also see that multiplying on the left by an invertible $(d+1) \times (d+1)$ matrix has the effect of applying a common projective transformation to the points of A. Thus two

$(d+1)\times m$ matrices represent equivalent objects if we can convert one to the other by elementary row operations and non-zero column scaling. Using a slightly modified version of Gaussian elimination, we can compute a fundamental set of invariants for a matrix representing m points in space (i.e. in $\mathbb{P}^3$) by moving the first 5 points to a "standard position."

$$\begin{bmatrix} a_{1,1}\cdots a_{m,1} \\ a_{1,2}\cdots a_{m,2} \\ a_{1,3}\cdots a_{m,3} \\ a_{1,4}\cdots a_{m,4} \end{bmatrix} \sim \begin{bmatrix} * & 0 & 0 & 0 & 1 & p_{6,0}\cdots p_{m,0} \\ 0 & * & 0 & 0 & 1 & p_{6,1}\cdots p_{m,1} \\ 0 & 0 & * & 0 & 1 & p_{6,2}\cdots p_{m,2} \\ 0 & 0 & 0 & * & 1 & p_{6,3}\cdots p_{m,3} \end{bmatrix}.$$

Similarly for points in $\mathbb{P}^2$ we move the first 4 points to standard position:

$$\begin{bmatrix} a_{1,1}\cdots a_{m,1} \\ a_{1,2}\cdots a_{m,2} \\ a_{1,3}\cdots a_{m,3} \end{bmatrix} \sim \begin{bmatrix} * & 0 & 0 & 1 & q_{5,0}\cdots q_{m,0} \\ 0 & * & 0 & 1 & q_{5,1}\cdots q_{m,1} \\ 0 & 0 & * & 1 & q_{5,2}\cdots q_{m,2} \end{bmatrix}$$

In effect we are moving P_1 to the point in projective space with homogeneous coordinates $[1:0:0:0]$, P_2 to $[0:1:0:0]$ and so on, with P_5 moving to $[1:1:1:1]$. This presupposes that P_1, P_2, P_3, P_4 are not co-planar and that P_5 is not co-planar with any three of P_1, P_2, P_3, P_4, i.e. we are assuming that $P_1, \ldots, P_5$ are in general position. Actual generators for the field of invariant rational functions under projective transformations on the moduli space of m-tuples of points in $\mathbb{P}^3$ are $\left\{\frac{p_{i,0}}{p_{i,3}}, \frac{p_{i,1}}{p_{i,3}}, \frac{p_{i,2}}{p_{i,3}} \quad \text{for } i=6,\ldots,m\right\}$. (Strictly speaking, to have a reasonable moduli, one must restrict to "stable" or "semi-stable" m-tuples.[9])

These particular functions provide coordinates on an open subset of the quotient space of m-tuples modulo the action of PGL_4, the group of projective transformations on $\mathbb{P}^3$. This open subset $\Gamma_{3,\ldots,3}$ is defined by the non-vanishing of the coordinates $p_{i,3}$ for $i=6,\ldots,m$ in addition to the general position assumption on $P_1,\ldots,P_5$. We can similarly define $\Gamma_{d_6,\ldots,d_m}$ to be the open subset defined by the general position assumption on $P_1,\ldots,P_5$ along with the non-vanishing of p_{i,d_i} for $i=6,\ldots,m$ and analogously arrive at affine coordinates on each $\Gamma_{d_6,\ldots,d_m}$.

For reasons of numerical stability and in order to avoid imposing any assumptions on the points aside from the general position of $P_1,\ldots,P_5$, we work with the homogeneous coordinates $p_{i,j}$ rather than with any particular set of affine coordinates (ratios of the $p_{i,j}$).

If some other subset of five points from among $P_1,\ldots,P_m$ were in general position, while $P_1,\ldots,P_5$ were not, we would simply work in a different open subset of the quotient manifold with the invariant coordinates on that open subset; making use of the obvious change of coordinates to adjust our formulas below.

The object-image equations for this case were derived for the affine coordinates above in,[4] and reformulated for homogeneous coordinates in.[1]

THEOREM 3.1. *If $Q_1,\ldots,Q_m$ are the images of $P_1,\ldots,P_m$ under a perspective transformation then the relations $E1_j=0$ and $E2_{6,k}=0$ hold for $j=6,\ldots,m$ and $k=7,\ldots,m$, where*

$$\begin{aligned} E1_j \;=\; & (-p_{j,0}p_{j,1}+p_{j,0}p_{j,3})q_{5,0}q_{j,2} + \\ & (-p_{j,0}p_{j,3}+p_{j,2}p_{j,0})q_{j,1}q_{5,0} + \\ & (-p_{j,2}p_{j,1}+p_{j,3}p_{j,1})q_{5,1}q_{j,0} + \\ & (p_{j,2}p_{j,1}-p_{j,2}p_{j,3})q_{j,0}q_{5,2} + \\ & (-p_{j,3}p_{j,1}+p_{j,0}p_{j,1})q_{5,1}q_{j,2} + \\ & (-p_{j,2}p_{j,0}+p_{j,2}p_{j,3})q_{j,1}q_{5,2} \end{aligned}$$

$$\begin{aligned} E2_{6,k} \;=\; & (-p_{k,1}p_{6,0}+p_{k,3}p_{6,0})q_{5,0}q_{6,2}q_{k,2} + \\ & (-p_{k,3}p_{6,0}+p_{k,2}p_{6,0})q_{5,0}q_{6,2}q_{k,1} + \\ & (-p_{k,2}p_{6,3}+p_{k,3}p_{6,2})q_{5,2}q_{6,0}q_{k,1} + \\ & (p_{k,1}p_{6,0}-p_{k,1}p_{6,3})q_{6,2}q_{5,1}q_{k,2} + \\ & (p_{k,2}p_{6,3}-p_{k,2}p_{6,0})q_{5,2}q_{6,2}q_{k,1} + \\ & (p_{k,1}p_{6,3}-p_{k,1}p_{6,2})q_{6,0}q_{5,1}q_{k,2} + \end{aligned}$$

$$(p_{k,1}p_{6,2} - p_{k,3}p_{6,2})q_{5,2}q_{6,0}q_{k,2}$$

Furthermore for any $k \geq 7$ we can interchange the roles of the indices k and 6 in $E2_{6,k}$ (i.e. use $E2_{k,6}$ instead $E2_{6,k}$). Since these equations are all of a fixed size, we can test the object/image correspondence in time linear in m. Looking carefully at them one sees that no variable $p_{j,i}$ or $q_{j,i}$ appears in any term with degree higher than 1. This fact will be very useful when setting up our indexing schemes because it allows us to determine if the equations can have solutions in coordinate boxes:

PROPOSITION 3.2. *If F is a real valued multi-linear function in the variables $x_1, \ldots, x_n$, and $\mathcal{B}_\alpha^\beta := \{z \in \mathbb{R}^n | \alpha_i \leq z_i \leq \beta_i, \quad i = 1 \ldots, n\}$, then F achieves its extreme values over $\mathcal{B}_\alpha^\beta$ at the vertices of $\mathcal{B}_\alpha^\beta$.*

COROLLARY 3.3. *$F(\mathrm{x})$ is non-zero for all x in $\mathcal{B}_\alpha^\beta$ if and only if F is positive on all vertices of $\mathcal{B}_\alpha^\beta$ or negative on all vertices of $\mathcal{B}_\alpha^\beta$.*

Once the projective invariants (the q_{ij}) for an image are specified and the object invariants are treated as affine coordinates on a particular $\Gamma_{d_6,\ldots,d_m}$, the polynomials (dehomogenized with respect to the p_{id_i} $i = 6, \ldots, m$) become multi-linear functions of those $3(m-5)$ object invariants. Naively this tells us that to check if an equation $E1_j$ or $E2_{j,k}$ vanishes in a box in "object space" we need 2^{3m-15} evaluations. But looking carefully at the equations $E1_j$ we see that they each involve only the homogeneous coordinates for P_j, and thus in any of our affine charts involve only three variables. Since $E2_{6,k}$ and $E2_{k,6}$ involve only the homogeneous coordinates $p_{6,i}$ and $p_{k,i}$ they can involve at most 6 variables in any affine chart. In fact, not all of the $p_{6,i}$ and $p_{k,i}$ appear, and we can show that in any of the affine charts at least one of these two equations will involve 5 or fewer variables:

THEOREM 3.4. *Given known projective invariants for an image configuration of points and a box $\mathcal{B}_\alpha^\beta$ in some affine chart of object space, we can determine if $E1_j$ vanishes in $\mathcal{B}_\alpha^\beta$ with $8 = 2^3$ evaluations. For one of the two equations $E2_{6,k}$ and $E2_{k,6}$ this question can be answered with at most $32 = 2^5$ evaluations.*

Note that the $E1_j$ and $E2_{6,k}$ are homogeneous polynomials in the homogeneous coordinates. Thus while it does make sense to talk about them vanishing, their values are not defined when they don't vanish. In order to use the equations to measure how close an object and image come to matching, one must resolve this ambiguity by specifying a way to dehomogenize. In order to be able to use Theorem 3.4, we scale the "projective invariant" points (i.e. columns 6 through m of the "standard position" matrix) for the object so that all coordinates have absolute value less than or equal to one, and at least one coordinate in each column is equal to one. For the image invariants we are free to choose any particular normalization we please. In,[1] we investigated the performance of several choices of normalization in the presence of noise.

4. INDEXING/HASHING

In this section we will describe an indexing scheme based on the object-image formalism and then describe its implementation in the case of point configurations as described above. Here we are assuming a large data base of 3D models to be queried by a 2D image for all objects which match the image. For generality, we describe the database as storing points in some *data-space*, and querying by proximity to a *query-loci*. Two operations are supported: adding points (objects) to the data base and querying it.

The basic idea is that will we represent the objects in our data base as points in the appropriate space, and use the object-image equations to determine if they match or approximately match a query image. In other words, we will be looking for all objects on or near an algebraic subset of the object space defined by the vanishing of the equations in the previous section when the image invariants $q_{i,j}$ have been specified. The simplest way to do this is to simply store all the objects and check the equations for each object once the invariants for the query image are known. Our approach to avoiding the necessity of checking each and every object is a modified multidimensional access scheme. For a survey of multidimensional access methods we refer the reader to.[8]

The index takes the form of a tree whose nodes B (or boxes) consist of a *data-range*, denoted $range(B)$, representing a subset of the data-space and a data list $data(B)$. If B is a leaf of the tree, the data list will contain those data points in $range(B)$. For interior nodes B, the data list consists of boxes B_i whose ranges form a covering of $range(B)$ by subranges. While we do not require the subranges in B to be disjoint, we make the tacit assumption that the overlap is small and very few objects lie in more than one box of a subdivision.

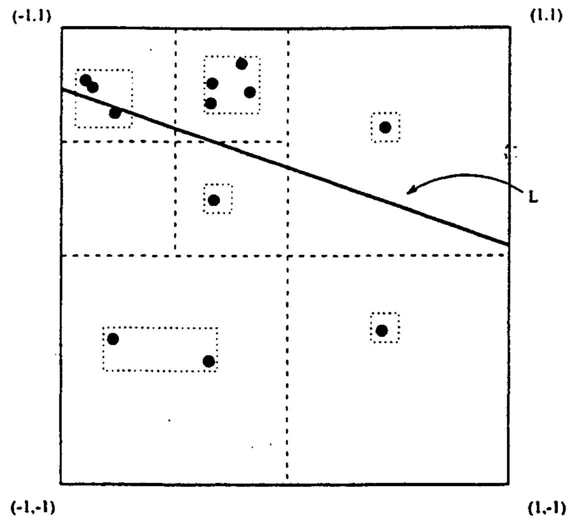

Figure 1. Indexing Example

The data base begins with a list of top-level boxes which cover the data-space and is built up by adding data-points one at a time. Adding a point P to a box B proceeds as follows: If P is in $range(B)$ then we add P to the list $data(B)$ if B is a leaf, or we add P to each node on the list $data(B)$ if B is interior. Whenever the data list of a leaf becomes too full it is made an interior node by subdividing its range and re-adding it's data points. Querying proceeds similarly, one starts a query at each top level box and descends to query children any time the query-locus passes through the boxes data region.

In order to implement this scheme, one must specify the data-space, query-loci, data-ranges as well as:

- A covering of the data-space by data-ranges.
- An algorithm for subdividing a range.
- An algorithm to determine if a data-point lies in a data-range.
- An algorithm to determine if the query-loci passes through a data-range.
- An algorithm to determine if the query-loci matches a particular data-point.

One embellishment that we do use in practice is to maintain two ranges, an exterior range used when adding points, and an interior range used when searching. The interior range is maintained so that it is always just big enough to hold the data points actually in the box or in its subtree.

As a simple example, we might take our data-space to be composed of points in the two dimensional box $[-1,1] \times$ $[-1,1]$, our query-regions to be lines, and our ranges to be boxes of the form $[a,b] \times [c,d]$. As a covering we just use the box $[-1,1] \times [-1,1]$ itself, subdivision can be accomplished by dividing a box with sides of length L into four boxes with sides of length $\frac{L}{2}$ and, of course, testing to see if a point lies in a box is trivial. As for querying, if the query line has the equation $Ax + By + C = 0$ we use the magnitude of $Ax + By + C$ as an indication of fit. The test to see if a query-region matches anything in data-range is easy if we recall that a linear functional achieves its maxima at the vertices of a box.

Our indexing scheme for object data bases is very similar to the example above. The query-regions will be subse of objects matching a given image. The data space is described by the 4^{m-5} charts mapping subsets of the fo $\Gamma_{d_6,\ldots,d_m}$ onto a copy of the coordinate box $B^{1,\ldots,1}_{-1,\ldots,-1} \subset \mathbb{R}^{3(m-5)}$. For data-ranges we take coordinate boxes B $\mathbb{R}^{3m-15}$. As in the example, determining if a data-point (an object) lies in a data-range is trivial, and we can sim take subdivision into 2^{3m-15} sub-boxes by dividing each edge in half. Finally Proposition 3.2 allows us to che query locus passes through a box efficiently by evaluating the individual terms $E1_j$ and $E2_{6,j}$ on a subset of the vertices

The performance of the indexing has been tested and the results will be reported on in future work.

5. APPENDIX: INVARIANTS REPRISED

In this section we will consider a more geometric approach to invariants and the object/image equations used in our indexing algorithm above. For simplicity we will consider the case of six points (x_i, y_i, z_i) $i = 1, \ldots, 6$ in $\mathbb{R}^3$

$$\begin{pmatrix} x_1 & x_2 & & x_6 \\ y_1 & y_2 & \cdots & y_6 \\ z_1 & z_2 & & z_6 \\ 1 & 1 & & 1 \end{pmatrix}$$

arrayed in a 4 by 6 matrix. Our only assumption will be that the points do not all lie in a plane, or equivalently that the rank of the matrix is 4.

Since we plan to use perspective projection, it is more appropriate to work in projective 3-space $\mathbb{P}^3$ where we use homogeneous coordinates for our points. (Hence the row of 1's in our matrix.) We could in fact carry out our discussion starting with any 4×6 matrix of rank 4 none of whose columns are zero

$$M = \begin{pmatrix} a_1 & & a_6 \\ b_1 & \cdots & b_6 \\ c_1 & & c_6 \\ d_1 & & d_6 \end{pmatrix}.$$

The ambiguity of homogeneous coordinates can be captured by allowing multiplication on the right by an arbitrary non-singular diagonal matrix

$$D = D(\alpha_1, \ldots, \alpha_6) = \begin{pmatrix} \alpha_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_6 \end{pmatrix} \qquad \alpha_i \neq 0.$$

Each α_i acts as a scale factor in the i^{th} column.

In effect, we are considering six points in $\mathbb{P}^3$ that are not coplanar. The principal invariance we are concerned with is invariance under the action of projective general linear group PGL_4 on $\mathbb{P}^3$. These transformations are represented by the action of 4×4 invertible matrices on the left. Notice that a scalar matrix acts trivially in this context since it does not change the point in $\mathbb{P}^3$ that each column represents.

Now viewing M as a linear map from $\mathbb{R}^6$ to $\mathbb{R}^4$, we have a kernel (null-space) K^2 which is a plane in $\mathbb{R}^6$ through the origin. Notice that if we multiply M on the right by a 4×4 invertible matrix A the new 4×6 matrix $M' = AM$ still has K^2 as kernel. Thus K^2 is a kind of "invariant" for M. The set of all two dimensional linear subspaces of $\mathbb{R}^6$ has a natural structure of a manifold, the Grassmanian $Gr(2,6)$, which is compact and 8 dimensional. We can regard K^2 as a point in $Gr(2,6)$.

Now we are working with 6 points in $\mathbb{P}^3$ and their homogeneous coordinates as the columns of M, and so we must confront the problem of scaling in the columns. As mentioned, this is equivalent to acting on M on the left by $D = D(\alpha_1, \ldots, \alpha_6)$ a diagonal matrix. If $\alpha_1 = \alpha_2 = \cdots = \alpha_6$ then the new 4 by 6 matrix $M' = MD$ has the same kernel K^2 as M, but if the α_i are not all equal the kernel changes. As $\alpha_1, \ldots, \alpha_6$ vary we get a family V^5 of two dimensional subspaces of $\mathbb{R}^6$, namely the kernels $K^2(\alpha_1, \ldots, \alpha_6)$ which are functions of $\alpha_1, \ldots, \alpha_6$. V^5 (actually its Zariski closure) is a 5-dimensional toric subvariety of $Gr(2,6)$, $V^5 \subset Gr(2,6)$, it is an invariant of our 6 points in $\mathbb{P}^3$ under the action of PGL_4.

The fact that V^5 has codimension 3 in the eight dimensional Grassmannian is no accident. Three is precisely the number of independent fundamental invariants for 6 points in $\mathbb{P}^3$ under projective linear transformations.

Using the Plücker embedding of $Gr(2,6)$ in $\mathbb{P}^{14}$ we can find equations for V^5 in the Plücker coordinates (the determinants of the fifteen 4 by 4 minors of M).

Now the key point. Suppose we project into the plane $\mathbb{P}^2$ via perspective projection. This can be achieved by premultiply our 4×6 matrix M by a 3×4 matrix of rank 3. The resulting matrix $N = TM$ has columns representing the homogeneous coordinates of the images of our six points in some coordinate system on the plane $\mathbb{P}^2$.

N has a kernel which is a 3-dimensional subspace H^3 of $\mathbb{R}^6$. The key observation is that, if K^2 is the kernel of M and $N = TM$, then the kernel H^3 of N contains K^2! Of course this is too simple. The real requirement is that H^3 contain $K^2(\alpha_1, \ldots, \alpha_6)$ for some choice of the α_i, i.e. the curve in $Gr(2,6)$ which is the Schubert cycle of planes through origin in $H^3 \subset \mathbb{R}^6$ must meet V^5. This can be expressed in polynomial terms and can serve as an alternative to our object image equations above. These equations will be global in the sense that no special position assumptions, beyond the points not being coplanar in 3D or collinear in 2D, are required. In addition this approach will be more robust in applications; something we will discuss in a future paper.

As a concrete example, consider a smaller problem. Let's take a 2 by 4 matrix of rank 2

$$M = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \end{pmatrix}$$

none of whose columns is zero. This corresponds to choosing 4 points in the projective line $\mathbb{P}^1$. A special case is

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

consisting of four points x_i for $i = 1, \ldots, 4$ in $\mathbb{R}^1$.

The Plücker coordinates of M are $(M_{12} : M_{13} : M_{14} : M_{23} : M_{24} : M_{34}) \in \mathbb{P}^5$ where $M_{ij} = \det \begin{pmatrix} a_i & a_j \\ b_i & b_j \end{pmatrix}$. The Plücker relation

$$M_{12}M_{34} - M_{13}M_{24} + M_{14}M_{23} = 0$$

holds and cuts out $Gr(2,4)$ in $\mathbb{P}^5$.

Now let K^2 be the kernel of M. The Plücker coordinates of K^2 can be shown to be $(K_{12} : K_{13} : K_{14} : K_{23} : K_{24} : K_{34})$ where

$$\begin{array}{ll} K_{12} = M_{34} & K_{23} = M_{14} \\ K_{13} = -M_{24} & K_{24} = -M_{13} \\ K_{14} = M_{23} & K_{34} = M_{12} \end{array}$$

As we vary K^2 by multiplying M on the right by a diagonal matrix

$$D(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = \begin{pmatrix} \alpha_1 & 0 & 0 & 0 \\ 0 & \alpha_2 & 0 & 0 \\ 0 & 0 & \alpha_3 & 0 \\ 0 & 0 & 0 & \alpha_4 \end{pmatrix}$$

we sweep out the "invariant" variety $V^3 \subset Gr(2,4) \subset \mathbb{P}^5$. In parametric form V^3 is given by

$$\begin{aligned} &(X_{12} : X_{13} : X_{14} : X_{23} : X_{24} : X_{34}) = \\ &\qquad (\alpha_3\alpha_4 M_{34} : -\alpha_2\alpha_4 M_{24} : \alpha_2\alpha_3 M_{23} : \alpha_1\alpha_4 M_{14} : -\alpha_1\alpha_3 M_{13} : \alpha_1\alpha_2 M_{12}). \end{aligned}$$

Of course the Plücker relation

$$X_{12}X_{34} - X_{13}X_{24} - X_{14}X_{23}$$

holds. To describe V^3 in $Gr(2,4)$ we need one additional equation. This is of course equivalent to usual the cross ratio!

Bibliographic number : C- 00- F05582

Title: Les dossiers de l'après- paix: processus de paix; sécurité: de nouveaux types de conflits.

Author(s) : KHEIR E.

Affiliation : (FR)

Type of document : Publication en série

Language : FRE

Country of origin : FR

Journal type: ARABIES (FR).

Information source : NO. 160; PP. 26- 30; 3 Phot.; DP. 2000/04

ISSN : 0983- 1509

Deposit : 05; P2901

Abstract: La signature d'un traité de paix syro- israélien et israélo- libanais va accélérer les changements dans la configuration géostratégique régionale et provoquer de nouveaux types de conflits. Quels seront les conséquences d'une telle "paix complète" dans la région? Les sous- titres sont: - le processus de paix: "une longue histoire"; les années charnières; - la conférence de Madrid ou la stratégie américaine raffinée: les structures américaines de la paix, Etats- Unis d'Europe, les autres nouvelles formes de menaces; - progrès et blocage des multilatérales; - les effets d'une paix complète: la conclusion d'une paix avec tous les voisins va permettre à Israel d'assurer avec les Etats- Unis, un role régional important. La manière de régler la question de l'eau entre Israel et la Syrie va affecter certainement la Turquie. Des axes pourraient se développer entre différents Etats au détriment d'autres.

AN INFO/PH

COSATI classification : 15 03; 05 04

NCC 15/03/00; 05/04/00

Descriptors: SECURITE INTERNATIONALE*; GEOSTRATEGIE*; PAIX; TRAITE INTERNATIONAL; ISRAEL; SYRIE; LIBAN; TURQUIE; EGYPTE; HISTOIRE; STRATEGIE POLITIQUE; ETATS UNIS; EUROPE; MENACE